Analysis of collisions in wireless sensor networks

Vighio, Muhammad Saleem; Ravn, Anders Peter

Published in:
21st Nordic Workshop on Programming Theory (NWPT 2009), Lyngby, Denmark, 14-16 October 2009

Publication date:
2009

Document Version
Publisher's PDF, also known as Version of record

Analysis of collisions in wireless sensor networks

Muhammad Saleem Vighio and Anders Peter Ravn

Department of Computer Science, Aalborg University,
Selma Lagerlöfs Vej 300, DK-9220 Aalborg East, Denmark.
{vighio,apr}@cs.aau.dk

Abstract. We present a model of the LMAC protocol for wireless sensor networks in order to analyse collision probabilities for nodes selecting a time slot. We consider fully connected topologies consisting of 3 and 4 nodes, and use probabilistic Uppaal (Uppaal PRO 0.2) for modelling and analysis. The results show that if the waiting time of nodes is increased before the selection of a time slot then the number of collisions decrease and vice versa. However, the results, which are exact due to the model checking technique, indicates that there is only a small loss by using optimistic wait times. This confirms the previously known results which have been reached through simulations. Based on our analysis results, we propose an optimistic choice of a network set-up depending on the probability of waiting times. Besides this, based on the probability of collisions we compare the results of the number of nodes to the number of time slots and suggest a network model for better performance and reduced cost.

1 Introduction

Network protocols are distributed algorithms and thus they are hard to comprenehnd and to validate. Furthermore they have much control and little data, therefore model checking has been very successfully used to verify this class of algorithms. Indeed, Holzman's book [5] shows that these algorithms were driving development of model checking. Since then model checkers for timed protocols have come along and been successful. With wireless sensor netoworks, new protocols appear, because they suffer from collisions caused by nodes transmitting data at the same time through the transmission medium. Therefore, efforts are taken at the MAC layer to reduce or minimize collisions. A characteristic example is the LMAC protocol [10] which is modelled and analyzed in [3] with the timed automata model cheker Uppaal [1]. Here, it is shown that the protocol is correct for an exhaustive set of communication topologies for up to 5 nodes. The developed automata usee non-deterministic transitions to model probablistic choices in the real protocol. It means that since the protocol is symmetric for each node, it is clear that the non-deterministic version cannot be proven to terminate with a successful slot allocation, cf. the impossibility results in [4]. We have taken up the challenge to rework the timed automata models to some that use probablistic choice. It has required simplification of the state space in order to accomodate the additional state space requirements of avaliable probabilistic model checkers. In some intial work, we used PRISM [8], but were not successful in getting conclusive results. This was probably caused by differences in the communication primitives used in Uppaal and PRISM and diffculties of encoding data structures. With the recently released version of Uppaal we have been successful in getting models that produce meaningful results. The contributions of this paper are thus: 1. a LMAC protocol model suitable for analysis with a probabilistic model checker, and 2. exact results that support the evidence provided by simulations in previous work [3].

Related work

This work is directly inspired by [3], where Fehnker, Hoesel and Mader, have presented modelling and verification of the LMAC protocol using Uppaal. The authors provide verification results of all possible connected topologies consisting of 4 and of 5 nodes. The property of main interest is the detection and resolution of collisions between the neighbouring nodes. We extend the work in [3] by investigating the probability of collisions for different waiting times for nodes before the selection of a new slot and propose an optimal network set-up policy.

In [9] Hoesel presents experimental results of different TDMA-based MAC protocols using the discrete event simulator OMNeT++. The author also compares the experimental results of SMAC [11], EMAC [2] and LMAC [10] protocols for wireless sensor networks. We found it very useful to read the work of Hoesel, in particular his investigation of different different waiting times for nodes before the selection of a time slot. The author uses simulation to evaluate the effects of waiting times. The simulation consists of one gateway node and 99 other nodes. We believe that simulation can deal with much higher numbers of nodes than model checking; but on the other hand, model checking traverses all possible scenarios which is hardly ever the case with simulation. For example executions that lead to collisions are in general a small subset of the full set of executions.

Overview

In Section 2, we describe the operation of the LMAC protocol while building the probabilistic model. Section 3 discusses model properties and verification results and finally Section 4 gives conclusion of the work.

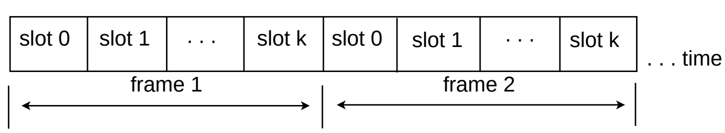

Fig. 1. A frame structure

2 The LMAC protocol and its probabilistic modeling

As stated in [3], the LMAC protocol is a scheduled based medium access control protocol for wireless sensor networks designed to work in a multi-hop, energy-efficient and self-configuring manner. Time is divided into time slots which are grouped into frames. Each frame has a fixed length of time slots adopted according to the network needs. Fig. 1 shows the structure of a frame. We use different lengths frame for different experimental results and each time slot takes two time units to complete. The purpose of different frame lengths is discussed in section 3.2. Every node can only send data (if it has some data to transfer) in its own time slot and can receive data during the time slots of other nodes in its transmission/reception range. If it is not in the reception range of other nodes it keeps its power consuming transceiver to standby to conserve energy. Before sending data a node always sends a short control message to obtain information about other nodes within a two-hop distance. A control message contains information about the slot occupied by the node itself and its first-order neighbours. With the information provided, receiving nodes determine which slots are unoccupied and can be used.

2.1 Timed automaton model of the protocol

As stated in [3], the LMAC protocol is divided into four phases. Initialization (I), Wait (W), Discover (D) and Active (A). We describe each phase with the help of the Uppaal models developed.

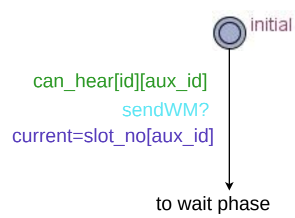

Fig. 2. Initial phase

Initialization phase When a new node joins the wireless sensor network it is unsynchronized and tries to detect other nodes. When it hears from a neighbouring node (*can_hear[id][aux_id]*) it synchronizes (*SendWM?*) and updates its current slot number to the slot number of the sender (*current=slot_no[aux_id]*) and moves to the *wait phase* This is shown in Fig. 2.

Wait phase A node can wait for at most three frames at *wait phase* depending on the weight of execution given to each transition. The model for wait phase is shown in Fig. 3. After the wait the node proceeds to the *discover phase*.

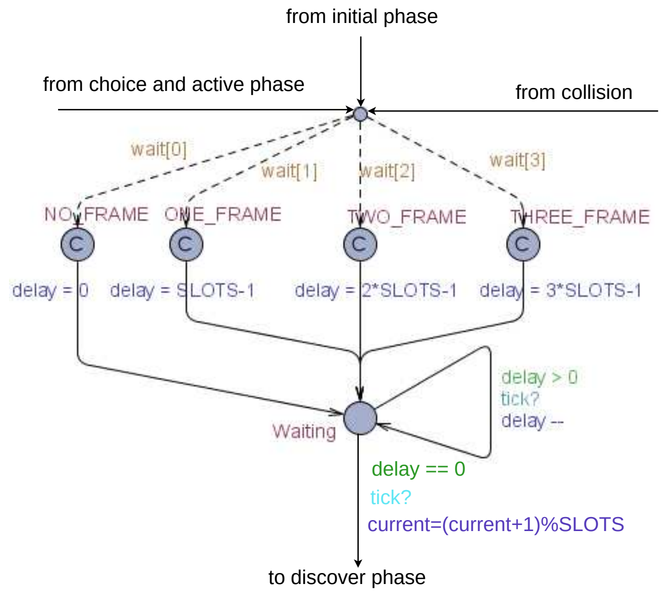

Fig. 3. Wait phase

During wait and later one, a node uses time to keep slots and frames globally synchronized. In the real protocol, local clocks are used, but in the model a global clock run by a scheduler is used. The scheduler simplifies the model and reduces the number of clocks and non-essential interleavings, while keeping the required behaviour of the protocol.

Discover phase At the *discover phase* (see fig. 4), the node collects first-order neighbourhood information during one complete frame. With this information the node determines which time slots in its second-order neighbourhood are free. The node then randomly selects a free time slot and moves to the *active phase*.

Active phase This phase deals with sending messages and is shown in Fig. 5. At the entry location f if the current slot number of a node is equal to its chosen slot number and has some first-order neighbouring nodes in its reception range *(current==slot_no[id] && first[id]!=0)* then it moves to location *ready*. At time unit one (t==*1*) it copies its *id* and *collision* information into global buffers *aux_id* and *aux_slot* and sends a message. However, if its current slot number is not equal to its chosen slot number then it will proceed to location *listening* and can either receive a control message form a neighbouring node or waits till the end of the time slot.

At location *rec_one*, a node may either receive a control message from a neighbouring node before the end of the time slot that reports a collision or it waits in this location till the end of the time slot. In the latter case the node checks if a collision has been reported and it is equal to its slot number. If it is

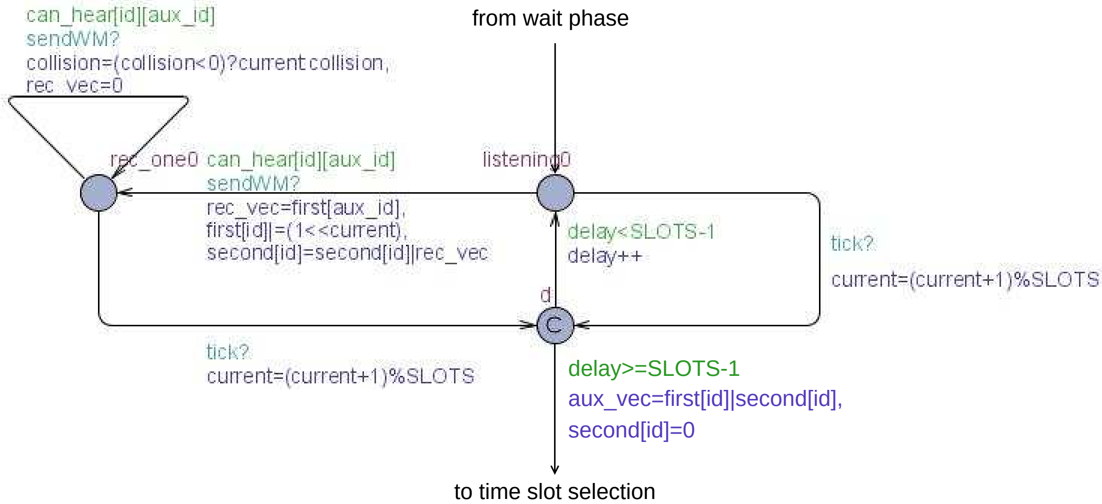

Fig. 4. Discover phase

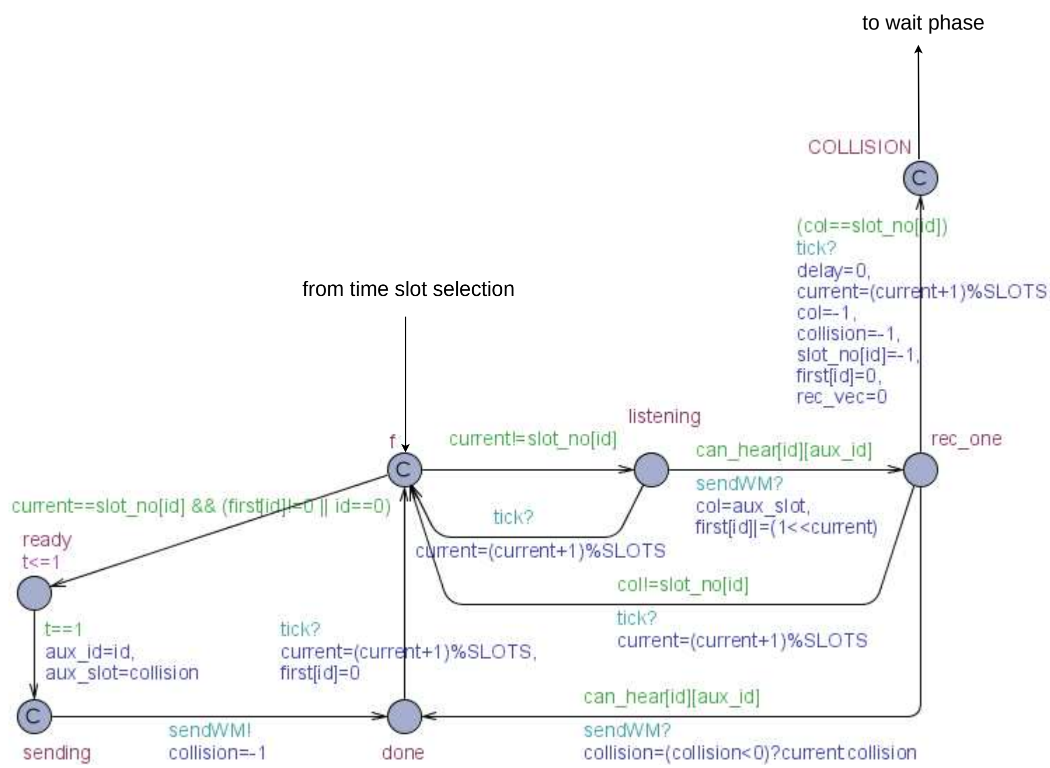

Fig. 5. Active phase

true *((col==slot_no[id]))* the node resets all the local information and moves to the *wait phase*. If the condition is false the node will proceed with the next slot.

2.2 State space considerations

From the specifications of the probabilistic modeling checking tools [7][6] and from our previous experience, we learnt that even a simpler probabilistic model can require a large amount of time and/or memory to construct and analyse. Therefore, in order to avoid state-space and verification time problems we have simplified the LMAC protocol model while preserving the required protocol behavior. Two further simplification tricks are performed:1) introducing most variables as meta variables and replacing non-deterministic time slot selection approach with a deterministic time slot selection function.

3 Model properties and verification results

For our verification results we develop network models consisting of 3 and 4 nodes. Each model consists of one gateway node which initializes the network. We use different lengths frames for our experiments with the number of slots greater than or equal to the number of nodes, because it is obvious that if the number of slots is less than the number of active nodes, the nodes will always collide, and thus the protocol has no chance of stabilizing. The property of main interest is to find the probability of collisions for different waiting times for nodes before selecting a time slot. We consider fully connected topologies only for our verification results, because in a fully connected network every node is connected to its neighbouring nodes and there is a higher probability of collisions than in a partially connected network. We also verify some safety, reachability and liveness properties to increase confidence in the correctness of the reduced model.

It is observed that, if the number of time slots is greater than the number of nodes then only few collisions will happen. However, more slots than the number of nodes increases network cost. Therefore for better performance and low network cost, we suggest an optimal number of slots to the number of nodes for a given network set-up.

3.1 Probability of collisions for different waiting times

Table 1. Probability of collisions w.r.t probability of different waiting times

x	**0**	**1**	**2**	**3**	4	5
Average waiting	**0.67**	**1**	**1.25**	**1.44**	1.6	1.72
Col. Prob.(3x4)	**0.25-0.5**	**0.18-0.51**	**0.14-0.53**	**0.11-0.56**	0.09-0.58	0.07
Col. Prob. (4x5)	**0.74-0.78**	**0.29-0.50**	**0.24-0.54**	**0.21-0.58**	0.19-0.63	0.17-0.67

Table 2. Probability of collisions w.r.t probability of different waiting times

y	0	1	2	3	4	**5**
Average waiting	2.33	2	1.75	1.56	1.6	**1.4**
Col. Prob.(3x4)	0	0.02	0.06	0.11-0.56	0.16-0.52	**0.21-0.50**
Col. Prob. (4x5)	0-0.14	0.11-0.31	0.18-0.39	0.23-0.68	0.28-0.64	**0.32-0.62**

As discussed in the *wait phase* of section 2.1, at our network set-up, a node may wait for at most three frames before selecting a time slot. This is shown in the second column of the Table 3. The selection of a particular wait is made on the weight given to each of the wait transitions *w0*, *w1*, *w2*, and *w3* as shown in Fig. 3. Higher value for any of the given transitions will execute that transition first than any of the lower weighted transitions. The transitions with equal weights are selected for execution randomly. We argue that if the waiting time of nodes increases then the number of collisions decreases and visa versa. This is shown in the verification results in Tables 1, 2.

The main aim of our verification results is to find the best case scenarios based on the probability of collisions and average waiting times. Table 1 shows the first case where weights for the transitions *w0*, *w1*, *w2* are kept constant and the weight for higher waiting transition *w3* is increasing and Table 2 shows the second case where the weight for lower waiting transition *w0* is increasing and the weights for other transitions are kept constant. The results in both the tables show that even with the higher probability of no waiting times we get few collisions and good average waiting time. Bold case lettered columns in Tables 1, 2 show some best case scenarios.

3.2 Comparing the probability of collision of the number of nodes to the number of time slots

Here we use a network model consisting of 3 nodes and compare the result on 3, 4, and 5. From the verification results obtained in Table 3, we observe that a network model of 3 nodes with 3 slots is not a better choice. We also found that there is a small difference on the probability of collisions for models with 4 and 5 slots. Therefore we concluded that a network model with one slot more than the number of nodes is a better choice for better performance and reduced cost.

4 Conclusion and future work

We have demonstrated a successful use of probabilistic model checking to support tuning of parameters for the probabilistic network protocol LMAC. The approach is fairly general and is likely to apply to other randomized algorithms. Yet, in contrast to simulation, model checking pays the price for exact results by accommodating small configurations only. However, in a practical wireless sensor setting, it is probably realistic to limit the number of nodes in a cluster such that

Table 3. Probability of collisions of the number of nodes to the number of slots

	Average waiting time	Waiting weights 0 1 2 3	Collision probability(3x3)	Collision probability(3x4)	Collision probability(3x5)
1	0	1 0 0 0	1	1	1
2	1.56	3 1 2 3	0.11-0.58	0.11-0.56	0.11-0.28
3	1.4	4 1 2 3	0.11-0.59	0.16-0.52	0.16-0.3
4	1.6	3 2 1 4	0.09-0.55	0.09-0.58	0.09-0.3
5	1.5	3 1 1 3	0.14-0.39	0.14-0.53	0.14-0.31

model checking is feasible. Here we expect that with some further simplification and a few tricks, we can get results for up to 5 nodes. An extension of the work would be to investigate the slot selection algorithm using a probabilistic model. However, this would probably have to be done for fixed waiting times, because a combined model might easily suffer state space explosion problems. An alternative modelling technique would use a continuous time distribution (exponential) to model waiting time.

References

1. Gerd Behrmann, Alexandre David, and Kim G. Larsen. A tutorial on UPPAAL. In Marco Bernardo and Flavio Corradini, editors, *Formal Methods for the Design of Real-Time Systems: 4th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM-RT 2004*, number 3185 in LNCS, pages 200–236. Springer–Verlag, September 2004.
2. T.Nieberg, S. Dulman, L.F.W. van Hoesel and P.J.M. Havinga. "Collaborative Algorithm for Communiation in Wireless Sensor Sensor Networks", Ambient Intellegence: Impact on Embedded Systems. November 2003.
3. Ansgar Fehnker, Lodewijk van Hoesel, and Angelika Mader. Modelling and verification of the lmac protocol for wireless sensor networks. In *Proc. IFM*, pages 253–272, 2007.
4. Faith Fich and Eric Ruppert. Hundreds of impossibility results for distributed computing. *Distrib. Comput.*, 16(2-3):121–163, 2003.
5. Gerard J. Holzmann. *Design and Validation of Computer Protocols*. Prentice Hall, 1991.
6. http://www.cs.aau.dk/ arild/uppaal probabilistic/documentation.shtml.
7. http://www.prismmodelchecker.org/manual/FrequentlyAskedQuestions/Main.
8. M. Kwiatkowska, G. Norman, and D. Parker. Probabilistic symbolic model checking with PRISM: A hybrid approach. *International Journal on Software Tools for Technology Transfer (STTT)*, 6(2):128–142, 2004.
9. L.F.W. van Hoesel. *Sensors on Speaking terms: Scheduled-based medium access control protocol for wireless sensor networks*. PhD thesis, June 2007.
10. Lodewijk F.W. van Hoesel and P.J.M. Havinga. A lightweight medium access control (lmac) protocol for wireless sensor networks: Reducing preamble transmissions and transceiver state switches. In *Proc. 1st International Workshop on Networked Sensing Systems*, 2004.
11. Wei Ye, John Heidemann, and Deborah Estrin. An Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *Proc. 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 1567–1576, 2002.